UI NAPRAVE

1. Zašto za mnoge USB naprave nije potrebno instalirati upravljački program već ih operacijski sustav može koristiti s postojećim?

ODGOVOR: Naprave su podjeljene po tipovima - za svaki tip postoji niz operacija koje trebaju podržavati. Npr. za spremanje podataka, za multimetdiju (video, audio, mirkofon), za komunikaciju

2. Opisati što rade slijedeće funkcije (u jezgri Linuxa), po čemu se međusobno razlikuju i kad ih koristiti?

mutex_lock_interruptible(&lock);
mutex_lock(&lock);
spin_lock(&key);

ODGOVOR:

-Sve su namijenjene za ostvarivanje kritičnog odsječka u jezgri.
-Prve dvije su blokirajuće - dretva se blokira ako je neka drugu u KO.
-Treća koristi radno čekanje.
-Blokiranje dretve prve, za razliku od druge, može biti prekinuto signalom.
-Blokirajuće pozive može se koristiti kad se *kod* jezgre izvodi u kontekstu dretve koja se može blokirati. Ne mogu se koristiti iz "atomarnih" okruženja, obrade prekida, alarma i slično
-Spin-lock se može koristiti iz svih okruženja, ali zbog negativnog učinka na performanse poželjno ga je koristiti samo tamo gdje su KO vrlo kratki.

3. U ostvarenju kruženog međusprenika "kfifo" veličina međuspremnika mora biti potencija broja 2 (N=2^X). Zašto? Što omogućava to ograničenje?

ODGOVOR: Takva veličina omogućava izbjegavanje modulo operacije. Umjesto njih koriste se binarne operacije. Umjesto X=(X+1) MOD N koristi se X = (X+1) & (N-1)

4. Prilikom uključivanja upravljakog programa u Linux, osnovne funkcije su:
alloc_chrdev_region(&dev_no, 0, 3, DRIVER_NAME);
cdev_add(&cdev, dev_no, 1);
Koju zadaću radi prava, a koju druga funkcija?

ODGOVOR: Prva traži (alocira) brojeve za napravu (major, minor)
Druga dodaje napravu (upravljači program naprave) u sustav.

5. Čemu služi sučelje request_irq i request_threaded_irq? Kako se ona međusobno razlikuju?

ODGOVOR:

- Prva služi za registraciju funkcije za obradu podataka.
- Druga pored toga može definirati i dretvu koja će se pokrenuti ako u obradi prekida ima više posla, ako sama obrada nije gotova u funkciji koja se izravno poziva pri prekidu.

6. Naprave se kroz OS koriste (uglavnom) istim sučeljima kao i datoteke. Međutim, neka od tih sučelja imaju puno veću ulogu pri komunikaciji s napravama. Navesti neke od takvih sučelja (barem dva) i kratko navedite čemu služe.

ODGOVOR:

1) ioctl - slanje "naredbi" napravi iz programa
2) poll - provjeravanje ima li koja naprava iz liste nešto novo
3) mknod - stvaranje datoteke za napravu
4) fsync - pražnjenje međuspremnika korištenog pri komunikaciji s napravom
5) aio_* - asinkrone operacije

7. Neke netrivijalne ulazno-izlazne operacije nije moguće napravit odjednom. Kako operacijski sustavi ostvaraju takve operacije, koje mehanizme koriste? Kratko ih opisati.

ODGOVOR:

1) **redovi zahtjeva** - stvara se zahtjev za operacijom i stavlja u red; obavljanje može potrajati, svaki puta kad se nešto napravi ažurira se takav zahtjev dok ne bude zgotovljen. Tek kad je zahtjev u potpunosti gotov dretva nastavlja s radom.

2) **jezgreni kontekst dretve** - ulaskom u jezgru iz programa aktivira se pridružena "jezgrina dretva"; u kontekstu te dretve obavlja se jezgrina funkcija; ona se može i blokirati u jezgri po potrebi - kao obična dretva, samo što se ova izvodi u jezgri.

8. (3) U komunikaciji s napravama često se koristi dvostruki međuspremnik (engl. *double buffer*). Prikazati korištenje takvog međuspremnika u primjeru u kojem OS treba poslati tri podatka napravi. Pretpostaviti da operacija naprave nad jednim podatkom iz međuspremnika traje 5 ms dok trajanje upisa podatka u međuspremnik od strane OS-a zanemariti.

t = 0 ms	OS stavlja 1. podatak u M1	naprava čeka
	OS stavlja 2. podatak u M2	naprava čita i obrađuje podatak iz M1
t = 5 ms	OS čeka	naprava završila s prvim podatkom
	OS stavlja 3. podatak u M1	naprava čita i obrađuje podatak iz M2
t = 10 ms		naprava završila s drugim podatkom
		naprava čita i obrađuje podatak iz M1
t = 15 ms		naprava završila s trećim podatkom
		naprava čeka

9. Velika većina komunikacija s napravama se danas obavlja korištenjem serijskih veza (npr. PCIe, USB, SATA). Koji su osnovni razlozi za prijelaz na serijsku vezu (npr. PCI->PCIe, PATA->SATA, USB umjesto paralelnih veza)?

ODGOVOR:

Mogu se postići veće brzine prijenosa jer se ne sinkroniziraju paralelni vodiči već samo susjedni bitovi na vodiču

10. Kod jezgrenih funkcija može se odvijati u različitim "kontekstima". Koji su to konteksti, kada se koriste i koje su njihove mogućnosti i ograničenja?

ODGOVOR:

1) **bez konteksta** - atomarno: prekidi, alarmi; kod se ne može blokirati, nema (jednostavnog) pristupa procesu - nije vezan uz korisnički proces

2) **u kontekstu jezgrine dretve**: prekidne dretve, odgođeni poslovi; kod se može blokirati, nema (jednostavnog) pristupa procesu - nije vezan uz korisnički proces

3) **u jezgrinom kontekstu procesa**: kad dretva pozove jezgrinu funkciju; kod se može blokirati, može se pristupiti procesu - izvođenje je povezano s korisničkim procesom

1 KORIŠTENJE NAPRAVA PREKO OPERACIJSKOG SUSTAVA

1. Kako se iz programa koriste naprave? Zašto baš tako?

- Iz perspektive operacijskog sustava (OS-a) upravljanje napravama obavlja se preko upravljackih sklopova na koje su naprave spojene.
- Naprave se iz programa koriste kroz usluge operacijskog sustava.
- OS upravlja napravama: inicijalizira ih, zna kako ih koristiti (preko "upravljačkih programa"),dozvoljava njihovo korištenje kroz funkcije koje OS nudi, rješava probleme "paralelnog korištenja" od strane više programa

2. Kojim sučeljima se mogu koristiti naprave? Koja su osnovna, a koja dodatna?

Naprave se mogu koristiti osnovnim operacijama i dodatnim operacijama
Osnovne: otvori, citaj/piši, zatvori. Dodatne operacije omogucuju odabir tražene informacije (pomak), upravljanje s više naprava odjednom (cekanje/provjera da se bar negdje nešto dogodi), slanje naredbi upravljačkom programu

-- **Osnovnim kao s datotekama:** open, close, read, write

-- **Dodatnim:** Iseek, select, poll, ioctl, fsynx,mkond,fcntl,readv….

- ➢ poll – provjera na više datoteka/naprava/*
- ➢ ioctl – slanje „naredbe” napravi

3. Što su to asinkrone operacije s napravama?

Asinkrone operacije su operacije koje se pokrenu te se ne čeka njihov kraj već se nastavlja sa nekim drugim poslom, a kraj operacije dozna se kasnije (npr. provjeravanjem statusa, primitkom signala, dodatnim pozivom).
To su funkcije koje počinju sa „aio_“

4. Navedite nekoliko mogućih podjela naprava ovisno o svojstvima, načinu spajanja, načinu komunikacije, smjeru podataka, ...

Fizičke dimenzije
Smjer: ulazna/izlazna/ulazno-izlazna
Brzina rada: brza/spora
Način spajanja: žično/bežično
Dohvat podataka: znak po znak/blokovi podataka
Smjer podataka: slijedno čitanje/svejedno čitanje
Način komunikacije: sinkrono/asinkrono

2 JEDNOSTAVNI MODEL OS-A ZA UPRAVALJNJE NAPRAVAMA

1. Ako zanemarimo način spajanja naprave, koji su osnovni načini upravljanja napravama (posluživanja naprava)?

Radno čekanje, prekidi, DMA

2. Opisati osnovna nacela upravljanja radnim cekanjem, prekidima te korištenjem sklopova s izravnim pristupom spremniku (DMA).

Radno čekanje – petlja u kojoj se čeka da naprava bude spremna
Prekidi – naprava javlja kad je spremna
DMA – naprava sama prenosi podatke u radni spremnik

3. U jednostavnom modelu jezgre ulazno-izlazne operacije može se pretpostaviti da naprava obavlja zadane joj naredbe slijedno. Stoga bi procesi koji su tražili takve operacije mogli biti u jednom uređenom redu, čekajući dovršetke svojih operacija. Koji su problemi ovog modela zbog specifičnosti stvarnih sustava? Kako se u stvarnim sustavima rješavaju takvi problemi?

- Naprave možda ne moraju posluživati zahtjeve po redu.
- Operacija tražena od procesa može zahtijevati više UI operacija, možda i s više naprava.
- U stvarnim sustavima koriste se redovi zahtjeva i jezgrin kontekst dretve.

4. Kako se rješava problem složenosti ostvarenja operacijskog sustava u stvarnim sustavima, npr. Linuxu?

Podjelom na podsustave i slojeve

3 MODEL NAPRAVA

1. Kako se može pristupiti napravama, tj. njihovim upravljačkim sklopovima?

- Napravama se najcešce može pristupiti korištenjem adresa ili preko portova. Adrese mogu biti prave ili virtualne.
- Adrese koje se koriste za komunikaciju s napravama su cesto mapirane u adresni prostor upravljackog sklopa - kontrolera na koji je naprava spojena.
- Ako OS koristi stranicenje onda te adrese treba mapirati u tablici prevodenja.

2. Zašto su stvarne arhitekture računala hijerarhijski građene, s premosnicama između različitih dijelova (sabirnica)?

Kako bi se sporije naprave stavile na sporije sabirnice, da ne utječu na performanse brzih naprava (koje bi inače morale čekati duge sabirničke cikluse sporijih naprava).

3. Nekim se napravama pristupa korištenjem adresa. Koji sve problemi zbog toga mogu nastati, tj. što treba "reći procesoru" u tom slučaju?

Ako OS koristi straničenje potrebno je adresu naprave mapirati u tablici prevođenja. Također je potrebno paziti da uvedene optimizacije u kodu ne narušavaju ispravnost rada naprave (npr. izmjena redoslijeda izvođenja operacija, korištenje priručnih spremnika)… Problem se rješava ubacivanjem posebnih instrukcija.

4. Zašto se koriste međuspremnici? Koju funkcionalnost obavljaju?

Međuspremnici se koriste radi povećanja učinkovitosti. Međuspremnik dozvoljava gomilanje podataka i praznog prostora te time omogućuje rad raznim brzinama.

5. Navesti vrste međuspremnika i opisati kako se oni koriste.

Međuspremnik za jedan podatak - jedna strana upiše podatak, pošalje signal da je podatak upisan, druga strana čita podatak, šalje signal da je podatak pročitan, repeat.

Dvostruki međuspremnik - dok naprava čita podatak iz prvog međuspremnika, OS piše u drugi međuspremnik. OS i naprava zatim rade sa suprotnim međuspremnicima itd. Tko piše, a tko čita ovisi o vrsti naprave. Ako zamijena nije istovremena onda netko čeka.

Kružni međuspremnik - princip rada sličan dvostrukom međuspremniku, ali s N međuspremnika. Podaci popunjavaju slijedno međuspremnik po međuspremnik, kada se dođe do kraja vraća se na vrh. Potrebno je pamtiti indekse zadnje upisanog i pročitanog podatka da nebi došlo do pisanja preko nepročitanih podataka.

6. Usporediti međuspremnik (engl. buffer) i priručni spremnik (engl. cache).

Međuspremnik/Buffer se koristi za prijenos podataka, a Prirucni spremnik/Cache za ubrzanje rada.

- U međuspremniku se spremaju stvarni podaci koji se brišu nakon njihovog čitanja. Podaci su samo na jednom mjestu u nekom trenuntku, nema kopija.
- U cache-u se sprema kopija podataka da bude bliža onom tko ih koristi. Ima više kopija
- Bolje je kod brzih naprava koristiti međuspremnik jer kopiranje podataka može usporiti operacije koje trebaju biti brže + međuspremnik omogućuje povećenje učinkovitosti ako naprave rade različitim brzinama.

Navesti vrste registara:

Upravljački, podatkovni, statusni registar

Čemu služi kružni međuspremnik s dvostrukim mapiranjem?

On rješava problem modulo aritemetike pri pisanju i čitanju preko granica. Isti segment memorije se dva put mapira. Veličina međuspremnika mora biti višekratnik veličine stranica kako bi se straničenjem prešla granica na iduću stranicu u virtualnom prostoru, a zapravo vratili na početak međuspremnika

a) prije dodavanja 13 elemenata:

```
--------------xxxxxxxxxx--------|--------------xxxxxxxxxx--------
               |          ^kms->ulaz
               kms->izlaz
```

b) nakon dodavanja 13 elemenata:

```
yyyyy---------xxxxxxxxxxyyyyyyyyy|yyyyy---------xxxxxxxxxxyyyyyyyyy
              ^kms->izlaz                ^kms->ulaz
```

c) kada `izlaz` pređe veličinu međuspremnika, nakon čitanja 20 bajtova:

```
--yyy---------------------------|--yyy---------------------------
  |   ^kms->ulaz
  kms->izlaz
```

4 PRIMJERI NAPRAVA

1. Navesti osnovna svojstva komunikacije preko serijske veze (npr. RS232). – -

- Korite se 2 vodiča za prijenos podataka. Po jedan u svakom smjeru.
- Za upravljanje su potrebna još minimalno 2 sinkronizacijska vodiča RTS i CTS
- Prenosi se bit po bit
- Potrebno je sinkronizirati početak/kraj posebnim znakom (XON/XOFF/DLE)
- Jedinica podataka je znak (5-8bita). Znak je grupa uzastopnih bitova
- Za velike udaljenosti
- Primjer RS232, USB, PCIe

2. Navesti osnovna svojstva komunikacije preko paralelne veze.

- U jednom ciklusu se više bitova prenosi paralelno preko više linija.
- Sinkronizacija nakon svakog bita
- Koristi se za kratke udaljenosti

*Problemi: Previše linija je potrebno, puno smetnji, previše sinkronizacije ograničava brzinu.

- Primjer PCI, DB-25

3. Navesti nekoliko protokola koji koriste serijsku i nekoliko koji koriste paralelnu vezu.

serijski: PCIe, SATA, USB, RS-232
paralelni: PCI, PATA , LPC, LPT,

4. Usporediti serijsku i paralelnu komunikaciju. Koje su prednosti serijske?

Serijska komunikacija je brža od paralelne jer koristi manje sinkronizacije. Serijska omogućava rad na većim udaljenostima i manje su smetnje.

5. Opisati kako su USB naprave spojene u računalu: logički, fizički.

Logički: sve su naprave na zajedničkoj sabirnici
Fizički: naprave su najčešće neizravno spojene preko mostova, čineći stablo s upravljačem u korijenu i napravama u listovima

6. Opisati osnovni način rada (komunikacije) naprave koja je spojena na USB priključak.

- Svaka naprava nakon spajanja dobiva adresu 7-birni broj. Sve naprave su spojene na upravljač koji ih periodički proziva imaju li nešto za javiti.
- Komunikacija se dijeli u okvire, svaki okvir je sastavljen od jedne ili više transakcija, a svaka transakcija od nekoliko paketa.

Svaki se paket (na fizickoj razini) sastoji od tri dijela:

1. sinkronizacijskog zaglavlja (SZ, sync)
2. tijela paketa (paket podatkovne razine)
3. oznake kraja paketa (KP, End-of-Packet:EOP)

- Uobičajena komunikacije uključuje 3 paketa:

1. TOKEN IN/OUT – šalje upravljač i određuje smijer prijenosa.
2. prijenos paketa s podacima – U smjeru postavljenom u Tokenu. OUT-naprava, IN- upravljač
3. Sinkronizacijski paket (ACK) – Paket kojim se određuje uspješnost prijenosa

7. Što su to okviri, transakcije, paketi u kontekstu protokola USB?

Komunikacija je podijeljena u okvire (engl. frame). Svaki okvir se sastoji od oznake pocetka okvira PO (engl. Start of Frame – SOF) te jedne ili više transakcija. Svaka transakcija se sastoji nekoliko paketa, ovisno o tipu transakcije.

Svaki se paket (na fizičkoj razini) sastoji od tri dijela:
1. sinkronizacijskog zaglavlja (SZ, sync)
2. tijela paketa (paket podatkovne razine)
3. oznake kraja paketa (KP, End-of-Packet:EOP)

8. Što je to adresa naprave, a što adresa funkcije naprave (kod protokola USB)?

-Adresa naprave se sastoji od same adrese naprave (7-bitovne adrese) te od rednog broja funkcije naprave (engl. end-point) koji je dodijeljen pri inicijalizaciji. Do 32 f-je po napravi.
-Jedna funkcija je zapravo operacija koju naprava može izvršiti.
- OS ne komunicira direktno s napravom već njenim funkcijama.

9. Za prijenos podataka USB-om preko protoka (engl. stream) koriste se cjevovodi: izokroni, prekidni i veliki. Opisati njihova svojstva i namjenu.

***Alternativno pitanje: Navedi 3 vrste prijenosa podataka cjevovodom za protok podataka:**

1) **Izokroni prijenos** – garantira propusnost, ali su mogući gubici – npr audio,video
2) **Prekidni prijenos** – Za brze odgovore na događaje – npr tipkovnica,miš
3) **Veliki prijenos** – Za prijenos veće količine podataka. Ne garantira propustnost i kašnjenja – npr. printer, skener

10.Zašto za mnoge USB naprave nije potrebno instalirati upravljačke programe, već ih operacijski sustav može koristiti s postojećim?

Protokol USB definira nekoliko klasa naprava korištenjem 8-bitovnog koda.
Naprave istih klasa nude sličnu funkcionalnost (npr. audio, miš, tipkovnica, hard disk, printer…).
Umjesto posebnih upravljačkih programa, mogu se koristiti uobičajeni programi specifični svakoj klasi.

11.Kako se prenose podaci preko PCIe? Koliko vodiča se koristi, je li moguć istovremeni prijenos u oba smjera?

- PCIe omogućava izravnu komunikaciju između naprava i kontolera.
- Podaci se prenose serijski, moguća je istosmjerna dvosmjerna komunikacija (full duplex),
- Koristi se 4 vodiča za svaku stazu tj 2 za svaki smjera. Može biti x1 ili više staza (x16,x32).
- Signal se prenosi u normalnom i invertiranom stanju, na taj način se smanjuju smetnje u sustavu.
- 128/130b kodiranje za PCIe 3. PCIe3 – 1GB/s po smjeru; za 16 staza (x16) = 32GB/s

12.Kako se prenose podaci u/iz memorije PCIe naprave?

- Svaka veza između naprava sastoji se između jedne ili više staza.
- Podaci se šalju u “paketima” na način da se paket razdijeli, a svaki dio paralelno putuje svojom stazom. Paket višekratnik od 32bita
- Svaki paket se potvrđuje ACK ili NAK porukama ako je bilo grešaka
- Dijelovi se na odredištu sastavljaju natrag u paket.
- Da ne dođe do zagušenja naprava javlja stanje međuspremnika.

13.Usporediti PCI i PCIe. Koje su prednosti sabirnice PCIe?

PCI - zajednička sabirnica (usko grlo)
- istovremena jednosmjerna komunikacija (half duplex)
- paralelna komunikacija, potreba za sinkronizacijom.
- najsporija naprava diktira tempo

PCIe - izravna komunikacija između naprava i kontrolera
- istovremena dvosmjerna komunikacija (full duplex)
- serijska komunikacija po stazama - brže od PCI

14.Koja je jedinica podataka (ne upravljačkih naredbi) koja se prenosi preko SATA protokola? Zašto nije proizvoljna veličina podataka?

SATA protokol koristi se za masivne spremnike memorije. Oni uglavnom adresiraju podatke u blokovima. Stoga SATA koristi fiksne veličine jedinica podataka pri prijenosu, tj. blokove, odnosno sektore, skupine blokova. Podaci se prenose serijski bit po bit uz diferencijsko signaliziranje kao i USB. 4 žice ukupno, 2 žice za svaki smijer. Full duplex

* USB 1 i 2 koriste 4 žice za spoj: +5V, GND, D+ i D-.
* D+ i D- za prijenos podataka u samo jednom smjeru u nekom trenutku. Signal se prenosi u normalnom i invertiranom stanju kako bi se smanjio utjecaj okoline.
* USB 3 i 4 imaju +2/3 para. Omogućuju istovremen obostran prijenos.
*Svaki prijenos započinje sinkronizacijskim zaglavljem i to brojem: 00000001
USB 1/2/3 kodiranje: 8b/10b | USB 3.1 : 128b/130b | USB 4: 64b/66b

*PCI: Adresa segmenta 16b; adresa PCI sabirnice 8b; adresa naprave 5b; adresa funkcije 3b

5 PODRŠKA ZA OSTVARENJE NAPRAVA U LINUXU

1. Unutar jezgre kod se može izvoditi u "različitim kontekstima". Koji su to i koja ograničenja postavljaju pojedini konteksti?

Jezgrin kod se može izvoditi u tri "okoline":

1. u jezgrinom kontekstu procesa (kad proces poziva jezgrinu funkciju)

- kada se jezgrina funkcija zove iz programa, npr. read() tada se pri ulasku u jezgru prelazi u jezgrin kontekst procesa
- umjesto dretve koja se izvodila u korisnickom nacinu rada, sada se aktivira jezgrina dretva koja izvodi jezgrinu funkciju.
  Ta dretva je "produžetak" korisnicke (ali sada u jezgri).
- jezgrina dretva može se i blokirati posebnim "unutarnjim" mehanizmima, sličnima "vanjskim" (semafori, monitori, ...)

2. u kontekstu jezgrine dretve (jezgrin proces)

- pri prihvatu zahtjeva zapocinje obrada prekida
- "obrada" koja je ovako zapocela ne smije u svom kodu imati blokirajuce pozive
- ako je potrebno više vremena za obradu, onda se posao obrade dijeli na dva dijela:
  1. neophodni dio – "top half"
  2. dodatni dio – "bottom half" - obavlja se naknadno, s dozvoljenim prekidanjem

3. bez konteksta dretve – "atomarno" (obrada prekida, alarmi, taskler, softirq)

2. Što treba napraviti/koristiti ako u jezgrinoj funkciji treba pristupiti adresnom prostoru procesa?

dohvat korisničkih podataka mora se obavljati korištenjem posebnih funkcija:

- copy_to_user
- copy_from_user

3. Obrada prekida naprave vrlo je bitan dio upravljanja napravom, ali može bitno utjecati i na svojstva sustava. Zbog čega? Koje mogućnosti u Linuxu stoje na raspolaganju za obradu prekida? Koja su njihova svojstva / kada ih koristiti?

Može bitno utjecati na svojstva sustava jer se izvodi u jezgrenom načinu rada I nije povezana s prekinutom dretvom pa se može odgoditi da je ta prekinua dretva izvodila neki važan posao pa bi trebalo što prije obaviti taj perekid I vratititi se u izvođenje dretve.

- ako je potrebno više vremena za obradu, onda se dijeli na 2 dijela:

1) Neophodni dio "top half" – onaj koji je započeo s obradom prekida ***(atomarni dio)***

2) Dodatni dio „bottom half" - obavlja se naknadno s dozvoljenim prekidanjem

Mogućnosti za dodatni dio:

a) Red poslova - workqueue

b) Višedretvena obrada prekida - threaded IRQs

c) Lagani prekid - softirq

d) Zadačić – tasklet

***** red poslova i višedretvena obrada se preporučuju. Oni se obavljaju u kontekstu posebnih jezgrinih dretvi i mogu biti odgođeni, blokirajući i sl.

* Lagani prekid i zadačić se izvode atomarno. Jednom započeti moraju biti dovršeni. Obavljaju se u kontekstu dretve koja obavlja takve poslove. NE SMIJU biti blokirani.

4. Što se smije a što ne koristiti u jezgri? Je li to ovisi o kontekstu u kojem se izvodi kod jezgre? Kako?

Ne smiju se koristiti sinkronizacijske funkcije, pristup adresnom prostoru procesa i realni brojevi

--to ovisi o kontekstu:

1. **U atomarnom kodu** (obradi prekida/top half, alarmima, taksletima itd):
    a. Ne smiješ nikakve fje koje mogu blokirati
    b. Nikakve koje traže kontekst dretvi (sem, monitori)
    c. MOŽEŠ funkcije s radnim čekanjem

2. **U Kodu s kontekstom** (u kontekstu procesa, WORKQUEUE, THREAD):
    a. Sve funkcije (interne jezgine) mogu jer worqqueu i threaded IRQs mogu biti blokirajući.

5. Što je to modul u kontekstu jezgre Linuxa? Čemu služi?

Upravljački program koji se dinamički učitava u jezgru te miče kada više nije potreban.

In general:
- upravljački program može biti na dva načina uključen u jezgru Linuxa:
    1. permanentno (statički)
    2. dinamički kao modul
- upravljački program može biti pripremljen zajedno s cijelom jezgrom Linuxa ili pripremljen kao "modul" koji se po potrebi, dinamički učitava u jezgru te miče kada više nije potreban

6. Navesti tri osnovne klase naprava u Linuxu.

tri su osnovne klase naprava (stvarnih i virtualnih) u Linuxu

1. **znakovne naprave** – character devices
    – naprave koje daju/primaju niz bajtova; npr. tipkovnica, miš, terminal, pisac
2. **blokovske naprave** – block devices
    – naprave kojima je jedinica podataka blok; najčešće naprave koje ostvaruju datotečne sustave
3. **mrežna sučelja** – network devices
    – naprave kojima je svrha ostvarenje komunikacije (koriste se iz mrežnog podsustava)

7. Koja je zadaća upravljačkog programa naprave?

- Upravljacki programi upravljaju stvarnim ili virtualnim napravama. On registrira svoje funkcije za neke dogadaje.
- Uobicajene takve funkcije su: otvari, zatvori, citaj, piši, pomakni, asinkrono citaj/piši, šalji upravljacke naredbe, mapiraj memoriju, zakljucaj,...

8. Koja su uobičajena sučelja koja znakovna naprava mora ostvariti, a da bi se uklopila u Linux? open, release, read, write, ioctl? Maybe. – ne piše u skripti

6. Primjer upravljačkih programa u Linuxu

TODO

Dodatna pitanja:
za sta se koristi alloc_chrdev_region, a za sta cdev_add

-zasto kfifo meduspremnik ima velicinu potencije broja dva
- cemu sluzi funkcija poll
- cemu sluzi funkcija ioctl

Opisati barem dvije sustavske funkcije za rad s napravama